D 8

```
LLL                  IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                  IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                  IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                     III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                     III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                     III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                     III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                     III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                     III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                     III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                     III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                     III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                     III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                     III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                     III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                     III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                     III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                     III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL      IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL      IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL      IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

N 13

```
**FILE**ID**OTSLUB
```

```
  000000    TTTTTTTTTT    SSSSSSSS  LL          UU      UU  BBBBBBBB
  000000    TTTTTTTTTT    SSSSSSSS  LL          UU      UU  BBBBBBBB
00      00      TT      SS          LL          UU      UU  BB      BB
00      00      TT      SS          LL          UU      UU  BB      BB
00      00      TT      SS          LL          UU      UU  BB      BB
00      00      TT      SS          LL          UU      UU  BB      BB
00      00      TT        SSSSSS    LL          UU      UU  BBBBBBBB
00      00      TT        SSSSSS    LL          UU      UU  BBBBBBBB
00      00      TT              SS  LL          UU      UU  BB      BB
00      00      TT              SS  LL          UU      UU  BB      BB
00      00      TT              SS  LL          UU      UU  BB      BB      ....
00      00      TT              SS  LL          UU      UU  BB      BB      ....
  000000        TT      SSSSSSSS    LLLLLLLLLL  UUUUUUUUUU  BBBBBBBB        ....
  000000        TT      SSSSSSSS    LLLLLLLLLL  UUUUUUUUUU  BBBBBBBB        ....


  SSSSSSSS  DDDDDDDD    LL
  SSSSSSSS  DDDDDDDD    LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
  SSSSSS    DD      DD  LL
  SSSSSS    DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
```

```
{ REQUIRE file for Logical Unit Block (LUB)
{ File: OTSLUB.SDL Edit: MDL2005
{
{****************************************************************************
{*                                                                          *
{*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
{*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
{*  ALL RIGHTS RESERVED.                                                    *
{*                                                                          *
{*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
{*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
{*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
{*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
{*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
{*  TRANSFERRED.                                                            *
{*                                                                          *
{*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
{*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
{*  CORPORATION.                                                            *
{*                                                                          *
{*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
{*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
{*                                                                          *
{*                                                                          *
{****************************************************************************
{
{ Author: T.Hastings
{ Change history:
{ [Previous edit history removed.  SBL 24-Aug-1982]
{ 1-048 - Give LUB$A_ASSOC_VAR another name LUB$L_ALQ, this is to hold the
{         allocation quantity for files in BASIC. FM 1-Oct-1980
{ 1-049 - Add a flag to request ANSI processing.  PLL 22-Jul-1982
{ 2-001 - Convert to SDL.  SBL 24-Aug-1982
{ 2-002 - Don't depend on names for sub-structures.  SBL 29-Sep-1982
{ 2-003 - Change aggregate name to LUB for better fieldset name.  SBL 26-Oct-1982
{ 2-004 - Add fields for RFA cacheing.  SBL 2-June-1983
{ 2-005 - add field to indicate FIELDing.  MDL 29-Mar-1984
{--

{+
{ The LUB contains OTS OWN data associated with a
{ particular logical unit which is needed between I/O statements.
{ OWN data which is needed for several calls within a single
{ I/O statement is allocated in the I/O Statement Block (ISB).
{ Data which is needed during a single call is always LOCAL.
{
{ LUB definition (-11 OTS equivalents are indicated in parens)
{ Symbols are of the form: LUB$t_symbol where t is
{ A,B,W,L,T,or V.
{-

MODULE $LUBDEF;
AGGREGATE LUB STRUCTURE PREFIX LUB$ ORIGIN end_of_lub;

{+
{ Define some constants that are used to set the organization field of the LUB
```

```
LIB

!+
! $

! M
! n
! f

! F



!-

MAC
```

```
{-

CONSTANT (
        ORG_SEQUE,  { Organization sequential
        ORG_RELAT,  { Organization relative
        ORG_INDEX,  { Organization indexed sequential
        ORG_TERMI,  { Organization terminal format
        ORG_VIRTU   { Organization virtual
        ) EQUALS 1 INCREMENT 1;

{+
{ Define the symbols for the special LUNs with negative numbers.
{-

CONSTANT (
        LUN_BPRI,   { Logical unit for BASIC PRINT
        LUN_INPU,   { Logical unit no. for BASIC INPUT
        LUN_BREAD,  { Logical unit for BASIC READ
        LUN_ENCD,   { Logical unit for FORTRAN ENCODE/DECODE
        LUN_READ,   { Logical unit no. for FORTRAN READ
        LUN_ACCE,   { Logical unit no. for FORTRAN ACCEPT
        LUN_TYPE,   { Logical unit no. for FORTRAN TYPE
        LUN_PRIN    { Logical unit no. for FORTRAN PRINT
        ) EQUALS -8 INCREMENT 1;

CONSTANT ILUN_MIN        EQUALS LUB$K_LUN_BPRI;  { Min LUN (for LUN table)
CONSTANT DLUN_MIN        EQUALS LUB$K_LUN_READ;  { Min default-OPEN LUN for FORTRAN
CONSTANT DLUN_MAX        EQUALS LUB$K_LUN_PRIN;  { Max default-OPEN LUN for FORTRAN
CONSTANT LUN_MIN         EQUALS 0;               { Min. explicit FORTRAN OPEN LUN
                                                 { BASIC limit is .GT. this.
CONSTANT LUN_MAX         EQUALS 119;             { Max. explicit or implicit LUN

{+
{ Define a symbol for the default value of the right margin.  This value
{ is used only by BASIC.
{-

CONSTANT D_MARGIN        EQUALS 72;              { default right margin for files

{+
{ Define a symbol for the maximum length of the prompt buffer.
{ This much space is allocated whenever a sequential file is opened
{ in case we are to prompt for input from it.
{-

CONSTANT PBUF_SIZ        EQUALS 80;              { Size of prompt buffer

{+
{ Lay out the storage of the LUB
{-

        CONSTANT NEG_BLN EQUALS . ;         { define negative length of LUB
        UBF ADDRESS;                        { RMS User Buffer.
        union_1 UNION;
            UNIT_STT3 WORD UNSIGNED;        { more flags
            UNIT_STT3_STRUCT STRUCTURE;
            NOECHO BITFIELD;            { If the BASIC function NOECHO has been done.
                                        { (applies to terminals only)
            ONECHR BITFIELD;            { If the BASIC function ONECHR has been done
                                        { (applies to terminals only)
            CCO BITFIELD;               { Cancel control O (BASIC function RCTRLO)
                                        { (applies to terminals only)
            FIND_LAST BITFIELD;         { 1 if last direct operation was FIND
                                        { (FORTRAN direct access)
            PTA BITFIELD;               { Purge type ahead
                                        { (applies to terminals only)
            AST_GUARD BITFIELD;         { (BAS-new) Used in I/O element transmit
                                        { to detect concurrence of ASTs using the
                                        { I/O data base.  If concurrence is detected
                                        { then the routine is repeated.
            CR BITFIELD;                { 1 if FAB$V_CR is set. (FORTRAN)
            FTN BITFIELD;               { 1 if FAB$V_FTN is set. (FORTRAN)
            PRN BITFIELD;               { (BAS-new) ≡1, PRN format and semantics for
                                        { Basic stream files.
                                        { 1 if FAB$V_PRN is set. (FORTRAN)
            NOMARGIN BITFIELD;          { (BAS-new) ≡1, the right margin for terminal
                                        { format files is infinite.
            USEROPEN BITFIELD;          { 1 if file opened with USEROPEN
            NOTSEQORG BITFIELD;         { 1 if file is not sequential organization.
                                        { Used by FORTRAN ENDFILE.
            ANSI BITFIELD;              { 1 if ANSI INPUT
            RFA_CACHE_ENABLE BITFIELD;  { 1 if RFA cacheing enabled (FORTRAN)
            FIELD_USE BITFIELD;         { 1 if FIELD stmt on this ch. (BASIC)
            fill_T BITFIELD LENGTH 1 FILL TAG $$;       { This many bits remain in this status word
            END UNIT_STT3_STRUCT;
        END union_1;

    BLS WORD UNSIGNED;                  { Mag tape block size, from FAB$W_BLS

{+
{ The following address, if non-zero, points to a routine to be called
{ just before the LUB is CLOSEd.  This is used by the BASIC File Array
{ support to write out the last buffer.
{-

    CLOSE ADDRESS;                      { Call here on CLOSE

{+
{ The following quadword is used to link the LUB to the LUB table
{ maintained by OTSCCB.
{-

    QUEUE QUADWORD UNSIGNED;            { Link for INSQUE and REMQUE instructions

{+
{ LUB Locations used by all User Data Formatted (UDF) level Procedures
{ which are: FOR$$UDF_{R,W}{F,U,L} and BAS$$UDF_{RW}_L
{-

    BUF_PTR ADDRESS;                    { (FOR-BLBUF) Adr. of next byte in buffer to be
                                        { filled or emptied with user data
    BUF_END ADDRESS;                    { (FOR-EOLBUF) Adr.+1 of last byte in buffer
```

```
                                        { to be filled or emptied with user data.
{+
{ This is the buddy pointer for BASIC I/O.  For all units except 0, it
{ should point to itself.  For unit 0, the Print CCB will point to the
{ Input CCB and vice versa
{-

    BUDDY_PTR ADDRESS;                  { pointer to the complementary CCB for Print and
                                        { Input.  Needed for recursive and continued I/O

{+
{ LUB Locations used solely by the input or output dependent
{ Formatted User Data Formatters .
{-

    BUF_BEG ADDRESS;                    { (FOR-LNBUF) Adr. of first byte in buffer (FOR-used
                                        { for T format).
    BUF_HIGH ADDRESS;                   { (FOR-TSPECP) Adr. of highest byte filled in
                                        { buffer during format processing (FOR-needed
                                        { because Tn format can move backwards).

{+
{ LUB locations set by OPEN, default OPEN, CALL FDBSET, or
{ DEFINE FILE, and checked on every I/O statement
{-

    ORGAN BYTE UNSIGNED;                { (BAS-IF.BLK-IF.TRF) File organization:
                                        { virtual block, sequential, relative, indexed
                                        { sequential or terminal format.
    BKS BYTE UNSIGNED;                  { Bucket size, from FAB$B_BKS
    LUN WORD;                           { Logical unit number (0:99)
                                        { Note: signed! Negative LUNS used for:
                                        { INPUT and PRINT
    union_1A UNION FILL;
        PRINT_POS LONGWORD UNSIGNED;    { (BAS-POSITN) printhead position
                                        { PRINT statements may end in a semicolon or
                                        { a comma requiring the printhead position to
                                        { be maintained to the next PRINT statement.
                                        { This is a longword because the longest string
                                        { (65K) may be put in the longest buffer (65K).
        RFA_CACHE_BEG ADDRESS;          { First allocated byte of RFA cache (FOR)
        END union_1A;
    union_1B UNION FILL;
        WAIT_TIME LONGWORD UNSIGNED;    { (BAS-WATIM) Wait time for a WAIT operation
        RFA_CACHE_PTR ADDRESS;          { Pointer to current RFA cache entry (FOR)
        END union_1B;
    IFI WORD UNSIGNED;                  { RMS internal file id, needed
                                        { until $CLOSE
    RBUF_SIZE WORD UNSIGNED;            { Record buffer size in bytes
                                        { Set by OPEN, default open, or DEFINE FILE.
                                        { Used to allocate record buffer at open.
                                        { Read by record level of abstraction
                                        { FOR$$REC_{R,W}{F,U,L}
    R_MARGIN WORD UNSIGNED;             { (BAS-new) the right margin for
                                        { a terminal format file.  The default is
                                        { 72 for terminal format files, set to terminal
```

```
LIB

!+
! P
! F

!**
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!*
!**

! E

! 1
! 1

!--
```

```
                                        { width otherwise.  Not the same as
                                        { buffer size because of embedded carriage con-
                                        { trol characters.  For terminal format files,
                                        { when the cursor position exceeds this value
                                        { the record is PUT.
    D_MARGIN WORD UNSIGNED;             { Default right margin.

    LANGUAGE BYTE UNSIGNED;             { The language that opened the LUN, as follows:
    CONSTANT LANG_MIN   EQUALS 0;       { Minimum language code
    CONSTANT LANG_NONE  EQUALS 0;       { None (probably not open yet)
    CONSTANT LANG_BAS   EQUALS 1;       { VAX-11 BASIC-PLUS-2
    CONSTANT LANG_FOR   EQUALS 2;       { VAX-11 FORTRAN-IV PLUS
    CONSTANT LANG_MAX   EQUALS 2;       { Maximum language code

    RFM BYTE UNSIGNED;                  { Record format, from FAB$B_RFM
    union_2 UNION;
        BAS_VFC WORD UNSIGNED;          { (BAS-new) fixed control block for carriage control
                                        { This is pointed to by the RAB so it is a part
                                        { of the RMS interface.  As a result, it is only
                                        { written to by the REC level.
        BAS_VFC_STRUCT STRUCTURE;
            BAS_VFC1 BYTE UNSIGNED;     { overlay first byte of VFC
            BAS_VFC2 BYTE UNSIGNED;     { overlay second byte of VFC
            END BAS_VFC_STRUCT;
        END union_2;

    union_3 UNION;
        ASSOC_VAR ADDRESS;              { Adr. of ASSOCIATEVARIABLE or 0 if none
                                        { Set by OPEN or DEFINEFILE.
                                        { LUB$V_ASS_VAR_L specifies word/longword

        ALQ LONGWORD UNSIGNED;          { Allocation quantity for files, also used in assoc_var
        END union_3;

    LOG_RECNO LONGWORD UNSIGNED;        { current or next Logical (or segmented)
                                        { record number for sequential access files
                                        { (needed for BACKSPACE of segmented records). Current or next
                                        { record number for
                                        { FORTRAN direct access files (0=1=first record)
    REC_MAX LONGWORD UNSIGNED;          { Direct access maximum record number
                                        { (RMS doesn't keep for Sequential organization
                                        { files). Set by DEFINE FILE or OPEN.
    FAB ADDRESS;                        { Address of FAB allocated by CALL ASSIGN,
                                        { CALL FDBSET, DEFINEFILE or OPEN.
                                        { 0 = ASSIGN, FDBSET, DEFINEFILE or
                                        { LUB not done.
                                        { NOTE: This field cannot move from an offset
                                        { of -24 due to Fortran compatibility routines
    RBUF_ADR ADDRESS;                   { Size in bytes of record buffer (includes
                                        { any FORTRAN information kept in the record)
    DID WORD UNSIGNED DIMENSION 3;      { three words to hold directory ID from OPEN
    RAT BYTE UNSIGNED;                  { The record attributes, from FAB$B_RAT.
    RSL BYTE UNSIGNED;                  { size of resultant name string (0 = no string allocated)
    RSN ADDRESS;                        { address of resultant name string

    union_4 UNION;
```

```
        UNIT_ATTR WORD UNSIGNED;        { (FOR-D.STAT) Unit attribute bits which are
                                        { needed between I/O statements.
                                        { NOTE: Some of these bits are in fixed
                                        { positions as noted.

        UNIT_ATTR_STRUCT STRUCTURE;
            OPENED BITFIELD;            { (FOR-DV.OPN) LUB has been successfully
                                        { opened by OPEN or default OPEN.
                                        { Cleared by CLOSE or error during OPEN
                                        { NOTE: cannot be moved from offset -4,0 due
                                        { to Fortran compatibility.
            IO_ACTIVE BITFIELD;         { (FOR--) An I/O statement is active on
                                        { this logical unit. Set to 0 on an error
                                        { or end of I/O list. Used to prevent recursive
                                        { I/O on the same logical unit.
            READ_ONLY BITFIELD;         { (FOR-DV.RDO) No writes will (can) be
                                        { done to this file.
                                        { Set by CALL FDBSET or OPEN 'READONLY'.
                                        { NOTE: cannot be moved from offset -4,2 due
                                        { to Fortran compatibility.
            OLD_FILE BITFIELD;          { (FOR-DV.OLD) Old (existing) file required,do
                                        { OPEN not CREATE.  Set by TYPE='OLD' or
                                        { FDBSET 'OLD'.
                                        { NOTE: cannot be moved from offset -4,3 due to
                                        { Fortran compatibility.
            DIRECT BITFIELD;            { (FOR-DV.DFD) FORTRAN direct access file.
                                        { Set by ACCESS='DIRECT' or DEFINEFILE.
                                        { Note: this bit is independent of RMS
                                        { file organization (Sequential or Relative).
                                        { Can not be moved from -4,4 unless
                                        { FOR$$IO_BEG is modified.
            SCRATCH BITFIELD;           { (FOR-DV.SCR) TYPE='SCRATCH' specified.
            DELETE BITFIELD;            { (FOR-DV.DEL) OPEN DISP='DELETE' specified.
                                        { Checked at CLOSE
            PRINT BITFIELD;             { (FOR-DV.SPL) OPEN DISPOSE='PRINT' causes
                                        { spooling at CLOSE.
            FORMATTED BITFIELD;         { (FOR-DV.FMP) File is FORTRAN formatted.
                                        { OPEN FORM='FORMATTED'
                                        { 0 = unspecified.
                                        { NOTE: Can not be moved from -4,8 unless
                                        { FOR$$IO_BEG is modified.
            UNFORMAT BITFIELD;          { (FOR-DV.UFP) File is FORTRAN unformatted.
                                        { 0 = unspecified. Set by DEFINE FILE or OPEN.
                                        { Note: LUB$V_FORMATTED and LUB$V_UNFORMAT
                                        { can both be 0 on default OPEN done for
                                        { END FILE since the format may be either.
                                        { Can not be moved from -4,9 unless
                                        { FOR$$IO_BEG is modified.
            FIXED BITFIELD;             { (FOR--) 1 = Record format is RMS fixed (FLR).
                                        { OPEN RECORDTYPE='FIXED'
                                        { 0 = Record format is RMS variable
                                        { (VLR or VLRM, i.e., VLR on Relative
                                        { Organization file.  OPEN RECORDTYPE='VARIABLE'
            SEGMENTED BITFIELD;         { (FOR--) Segmented (unformatted) records are
                                        { to be used. Otherwise only one
                                        { record (VLR or FLR) is to be read or
                                        { written with no segmented control info.
                                        { RECORDTYPE = 'SEGMENTED' in OPEN or TYPE not specified
                                        { for sequential unformatted file.
            ASS_VAR_L BITFIELD;         { (FOR-DV.AI4) ASSOCIATEVARIABLE is a longword
                                        { 0 = ASSOCIATEVARIABLE is a word or not
                                        { present. Set by OPEN or
                                        { DEFINE FILE.   See LUB$A_ASSOC_VAR
            APPEND BITFIELD;            { (FOR-DV.APD) File was opened ACCESS = 'APPEND'
                                        { also used as state bit (LOG_RECNO is undefined)
                                        { NOTE: cannot be moved from offset -4,13 due to
                                        { Fortran compatibilty.
            SEQUENTIA BITFIELD;         { 1 if ACCESS='SEQUENTIAL' (FORTRAN)
                                        { Note: Can not be moved from -4,14
                                        { unless FOR$$IO_BEG is modified.
            KEYED BITFIELD;             { 1 if ACCESS='KEYED' (FORTRAN)
                                        { Note: Can not be moved from -4,15
                                        { unless FOR$$IO_BEG is modified.
            END UNIT_ATTR_STRUCT;
        END union_4;

{+
{ Bits set by OPEN,default OPEN, CALL FDBSET,or
{ DEFINEFILE, and checked on every I/O statement
{-
    union_5 UNION;
        UNIT_STT2 WORD UNSIGNED;        { Second word of bits
        UNIT_STT2_STRUCT STRUCTURE;
            VIRT_RSN BITFIELD;          { indicates that RSN points to dynamic memory
                                        { not local storage
            ENDFILOPN BITFIELD;         { File was implicitly opened to do ENDFILE
                                        { When first I/O is done, there are a few
                                        { defaults which will be specified:
                                        { LUB$V_FORMATTED or LUB$V_UNFORMAT
                                        { LUB$V_SEGMENTED
                                        { Then LUB$V_ENDFIL_OPN is cleared.
            FORM_CHAR BITFIELD;         { (BAS-new) The last output element transmitter ended
                                        { in a comma or semicolon.
            OUTBUF_DR BITFIELD;         { (BAS-IF.WRT) the PRINT buffer already has
                                        { something in it and should be dumped
                                        { before continuing.  Set by BAS$$DO_WRITE
                                        { Used for same purpose by BASIC File Array support
            TERM_FOR BITFIELD;          { (BAS-IF.TRF) terminal format file on any unit including 0
                                        { Set by OPEN info from user. = 1, term-
                                        { inal format
            TERM_DEV BITFIELD;          { (BAS-IF.TRM) terminal device on any unit including 0
                                        { Set by OPEN info from RMS. = 1, term-
                                        { inal device
            FORCIBLE BITFIELD;          { (BAS-IF.FRC) forcible device on any unit including 0
                                        { Set by OPEN info from RMS. = 1, term-
                                        { inal or line printer device
            UNIT_0 BITFIELD;            { (BAS-new?) terminal device - unit 0
                                        { Set by Default OPEN for PRINT and INPUT
            VA_USE BITFIELD;            { (BAS-IF.VIR) marks a files first usage as a
                                        { virtual array.  Once used as virtual, it
                                        { cannot be used for block I/O.
            BLK_USE BITFIELD;           { (BAS-BIO) Marks a file's first use as block
```

```
                                      { I/O.  Once used as block I/O , it cannot be
                                      { used for virtual I/O.
            M_STREAM BITFIELD;        { (BAS-IF.CON) File is multistream connected.
            M_STR_C BITFIELD;         { File is (or was) connected to.
    {+
    { The following bit is set by CLOSE to indicate that the LUB should be
    { deallocated as soon as all recursive or nested I/O on it has
    { completed.  It is cleared (in effect) by OTS$$POP_CCB deallocating the
    { storage.  While it is set the LUN may not be OPENed since there is
    { I/O outstanding which should be allowed to fail.
    {-
            DEALLOC BITFIELD;         { Can deallocate this LUB
            SUBMIT BITFIELD;          { FORTRAN DISP='SUBMIT' if set.
            NULLBLNK BITFIELD;        { FORTRAN BLANK='ZERO' if clear,
                                      {  BLANK='NULL' if set.
            USER_RBUF BITFIELD;       { If 1, the record buffer was allocated
                                      { by the user, don't deallocate it at
                                      { CLOSE time.

            END UNIT_STT2_STRUCT;
        END union_5;

    CONSTANT LUB_LEN EQUALS :;        { Length of LUB

    end_of_lub BYTE FILL TAG $$;
    END LUB;

END_MODULE $LUBDEF;

{ End of file OTSLUB.SDL
```

0202 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY